Considerations for Isochronous Data Services over the Proximity-1 Space Link

Jay L. Gao*

Jet Propulsion Laboratory, Pasadena, CA 91109-8099

Future mission concepts for robotic and human explorations will involve a high level of real time control/monitoring operations such as tele-operation for spacecraft rendezvous and surface mobile platforms carrying life-support equipments. The timely dissemination of voice, command, and real-time telemetry for monitoring and coordination purposes is critical for mission success. It is envisioned that future missions will require a network infrastructure capable of supporting isochronous data services. The CCSDS Proximity-1 Space Link Protocol[1] could be used to carry isochronous traffic. This paper we will focus on the data link layer portion of the Proximity-1 protocol specification and analyze its jitter and performance for supporting isochronous applications. In particular we will focus on constrained scenarios where the protocol operates in full-duplex mode, carrying isochronous traffic in one direction and error-controlled traffic in the other direction. We analyze the impact of the strict priority scheme in Proximity-1 used to arbitrate channel access on the latency jitter of the isochronous traffic and the efficiency of the reliable data transfer. In general, jitter performance is driven by the loading of the acknowledgement traffic on the forward link. Under light loading condition, the upper-bound of the delay jitter is the transmission duration of an acknowledgement frame on the forward link; for higher loading scenarios, the maximum jitter is scaled up by the inverse of the residual bandwidth, i.e., the spare capacity available in the forward link after accounting for the acknowledgement traffic. We derive analytical expression on the maximum jitter and discuss its performance.

Nomenclature

ACK	=	acknowledgement for automatic repeat request
ARQ	=	automatic repeat request
CCSDS	=	Consultative Committee for Space Data System
COP-P	=	Command Operation Procedure – Proximity
EVA	=	Extravehicular Activity
FEC	=	Forward Error Correction
FIFO	=	First-In First-Out
Forward link	=	typically indicates out-going transmission from Earth to remote space assets; when using an orbiter for relay communications, the orbiter-to-remote spacecraft direction is the forward direction.
Return link	=	typically indicates in-coming transmission from remote space asset to Earth; when using an orbiter for relay communications, the remote spacecraft-to-orbiter direction is the return direction.
LSAM	=	Lunar Surface Access Module
Need_ACK	=	Boolean variable indicating need to update ARQ state to the sender
PLCW	=	Proximity Link Control Word - the data unit that carries acknowledgement/report of ARQ states
PLTU	=	Proximity Link Transmission Unit – carries data frame and the associated synchronization marker and parity bits
TDM	=	Time Division Multiplexing
T_{data_frame}	=	transmission time of a PLTU on the forward link (isochronous traffic)
λ_{data_frame}	=	data frame arrival rate of on the forward link (isochronous traffic)

* Technical Staff, Communications Networks Group, Telecommunications Architecture and Research Section, Jet Propulsion Laboratory, 4800 Oak Grove Drive, Mail Stop: 238-343, Email: Jay.L.Gao@jpl.nasa.gov.

ρ_{data}	=	loading of the isochronous traffic normalized to the capacity of the forward link
ρ_{Ack}	=	loading of the acknowledgement traffic normalized to the capacity of the forward link
R_{return}	=	data rate of the return link
$R_{forward}$	=	data rate of the forward link
N_{PLTU_ret}	=	number of bits in a return link PLTU
N_{PLTU_for}	=	number of bits in a forward link PLTU
N_{PLCW}	=	number of bits in a PLCW
T_{Ack}	=	transmission time of a PLCW on the forward link
T_{Need_Ack}	=	periodicity of the Need_ACK state being set to "True"
N	=	the minimum window size of the Proximity-1's COP-P retransmission procedure that maximize efficiency of the ARQ when there are no ACK losses and gaps
n	=	size of ACK frame gaps
η_{ARQ}	=	throughput of the return link, normalized to the error-free throughput of the ARQ algorithm
$T_{round\text{-}trip}$	=	round-trip time of the proximity-1 space link that incorporates propagation, transmission, coding, and process delays
$T_{def}(i)$	=	the delay experienced by the i-th ACK frame, from the moment the latest Need_ACK state is set to *True*, as a result of blocking by another ACK frame; when i=1, the delay is due to blocking by a data frame
D_{max_jitter}	=	the maximum jitter experienced by a data frame as a result of channel access contention with the ACK frames

I. Introduction

THE focus of this study is to analyze the performance of the CCSDS Proximity-1 Space Link Protocol[1] when carrying isochronous traffic such as voice, video, and command for tele-operations of networked space assets. No assumptions were made on the coding, synchronization, or physical layer. In particular, we examine a stressed scenario where isochronous traffic were intermixed with Proximity-1 acknowledgement frames generated by the reception of reliable data transfer on the reverse direction of the link. Isochronous services may become crucial for pre-cursor lunar exploration missions where a critical function is the effective tele-operation of robotic assets. Later phases of the exploration campaign may involve astronauts monitoring and controlling of space assets in-situ, while reliable data were returned back to the astronaut and mission control through a relay infrastructure.

The term 'isochronous' means "having equal time difference." Therefore, an isochronous communication system is one where the time difference between arrival and departure of any data is a constant. However, one can also extend the concept by describing the degree to which a system approximates the ideal isochronous system by quantifying the consistency of its latency characteristics, i.e., delay jitter. A 'deterministic' communication system deploying Time Division Multiple Access (TDMA) has high degree of isochronous property because the variability of latency is bounded by its TDMA frame duration. On the other hand, for systems that dynamically share resources the variations in latency could be larger, depending on the actual resource sharing mechanism deployed and the loading of the system. The CCSDS Proximity-1 protocol is not a deterministic system in the sense that it does not designate specific time slots to each type of data and control message, and the purpose of this analysis is to analyze the degree to which it can behave as an isochronous system.

The delay jitter metric measures the range of variation in the delay of data through the Proximity-1 protocol, but it does not imply that the inter-arrival times of successive data unit are necessarily constant. In this study we also ignore the queuing delay issue since buffering tends to eliminate the timeliness of real-time data traffic and is rarely deployed except at the minimum level necessary by the receiving application for the purpose of codec operation, which is not a link layer issue.

For terrestrial applications, jitter tolerance for toll-quality voice is typically set to about 20 msec. Besides jitter, the quality of voice is affected by other latencies due to processing, error correction coding, propagation delay, and transmission time of the data over the physical medium. Overall, the ITU-T G.114 recommendation prescribes a 150 msec one-way end-to-end delay as the maximum tolerable latency for two-way voice conversation before subjective experience of users becomes awkward (i.e., talking over each other). We can envision that for lunar exploration, the propagation delay between Earth and moon will present an insurmountable latency that requires special procedure for orderly conversations between multiple parties, i.e., token passing. However, for low-earth orbit operation and proximity communications around lunar and Mars region, the propagation factor is not as stringent and delay jitter will remain a key item in the latency budget.

II. Example Scenarios of Isochronous Services

As the CCSDS Proximity-1 Space Link protocol has proven its effectiveness through the success of the Mars Exploration Rovers, the concept of relay communication will soon be extended to the arena of human exploration. With the envisioned throughput for lunar exploration being very high, relay communications will again be the key to keeping the link budget at a reasonable level while meeting the coverage and throughput requirements. With the propagation delay between Earth and the Moon at about 1.5 seconds, tele-operation of lunar surface elements is a possibility, particularly for the early pre-cursor robotic missions, and isochronous communications between teams of Extra Vehicular Activity (EVA) astronaut is also anticipated in difference phases of the mission. In this section we provide a few examples of isochronous services arising for tele-operations of robotic platform, but they can also be generalized to voice communications.

In tele-operation, isochronous services is needed on the forward direction (from the controller to the remote asset) to ensure responsiveness of the remote assets to control/command. Reliability is typically provided via a combination of forward error correction (FEC) and link margin; retransmission is not desired since the latency may disrupt the timeliness of the command. In the return (feedback) direction, there could be a mixture of low rate motion-imagery, real-time status information, and science telemetry that requires stronger reliability via ARQ.

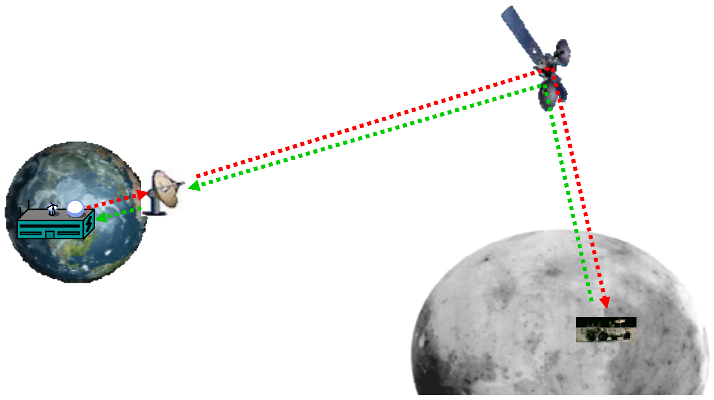


Figure 1: Tele-Operation of Lunar Asset

Figure 1 shows a potential pre-cursor mission scenario where a rover on the lunar surface is tele-operated from mission control via a relay orbiter. The relay orbiter improves the forward link capacity as well as providing capability of operating on the "far-side" of the moon. Here the Proximity-1 protocol is used over the orbiter-to-rover link. The long haul link from Earth to the orbiter could use other CCSDS standards. An extension of the scenario, as depicted in Figure 2, may involve additional lunar elements; say an EVA astronaut and a Lunar Surface Access Module (LSAM), which are receiving copy-traffic on the rover's return telemetry for situational awareness.

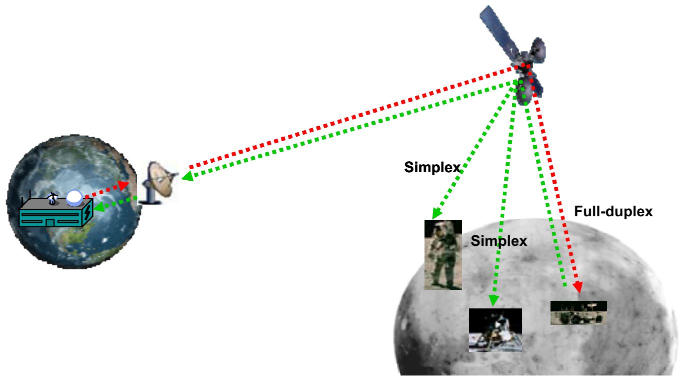


Figure 2: Tele-operation with EVA and LSAM monitoring progress

As shown in Figure 2, the CCSDS Proximity-1 Space Link protocol is simultaneously transmitting control directives to the rover as well as relaying return telemetry to the EVA astronaut and LSAM. The nominal operational scenario for Proximity-1 protocol is point-to-point communication. However, the framing structure allows for one-to-many transmission in a Time-Division Multiplexing (TDM) fashion, with the "copied" assets operating in a simplex, receive only mode. The transmitting side (the orbiter), can mark each Proximity-1 frame with the proper spacecraft ID so that each assets knows which frame to process and which frames to discard. On the receiving end, the physical layer handshaking and synchronization is completed between the relay orbiter and the rover, while the EVA astronaut and LSAM remain in simplex mode and listen to the orbiter's data stream and pull out frames addressed to them. This requires additional coordination in frequency channel to prevent interference.

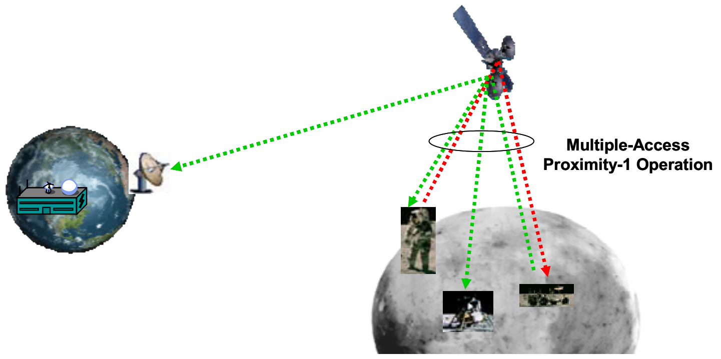


Figure 3: Over-the-horizon tele-operation over the lunar surface
(multiple access proximity-1 operation required)

Figure 3 shows a further variation of the remote tele-operation scenario where the controller is in-situ, in this case an EVA astronaut on the surface of the moon. The orbiter and the Proximity-1 protocol provide the over-the-horizon range extension so that the astronaut can control the rover without line-of-sight. LSAM and mission control may both monitor the return telemetry. In this case, the proximity-1 protocol is required to operate in multiple-access mode, transmitting to and receiving from multiple assets simultaneously. Since this scenario depends on extensions/modifications to the current standard, we will not consider it in this study. It should be noted that the envisioned usage of Proximity-1 is not limited to the tele-operation scenarios. Support for voice communication may also have similar, if not tighter, latency and jitter requirements and similar topology as described above. The choice of these particular scenarios are only motivated by their near term applicability to pre-cursor lunar exploration missions and serve to motivate this analysis.

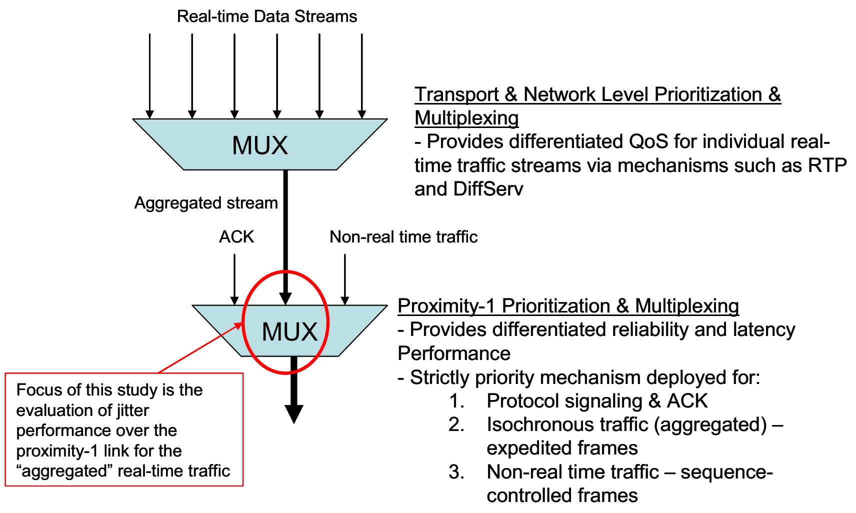
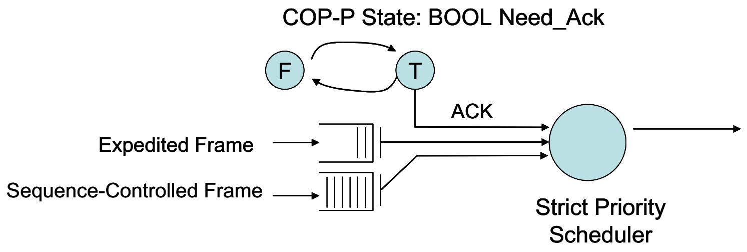

Real-time Data Streams
MUX
Transport & Network Level Prioritization & Multiplexing
- Provides differentiated QoS for individual real-time traffic streams via mechanisms such as RTP and DiffServ
Aggregated stream
ACK
Non-real time traffic
MUX
Proximity-1 Prioritization & Multiplexing
- Provides differentiated reliability and latency Performance
- Strictly priority mechanism deployed for:
1. Protocol signaling & ACK
2. Isochronous traffic (aggregated) – expedited frames
3. Non-real time traffic – sequence-controlled frames
Focus of this study is the evaluation of jitter performance over the proximity-1 link for the "aggregated" real-time traffic

Figure 4: Prioritized Data Handling of Real-time Data Streams

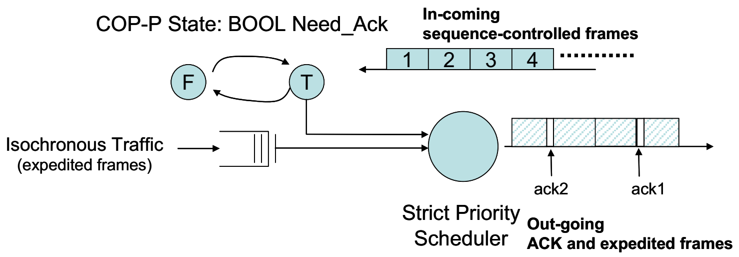
Figure 4 identifies the scope of our analysis. It is anticipated that in reality, many independent streams of isochronous data streams are multiplexed over a single link layer. It is assumed that individual data stream maybe distinguished and handled by the network and transport layers as traffic classes by Differentiated Services (DiffServ), as 'flows' by Integrated Service (InteServ), or by their association with transport layer 'ports.' However, below the network layer, the traffic classes, flows, or ports is further aggregated into broader link layer categories, either as expedited or sequence-controlled which is distinguished by their reliability and latency characteristics. Therefore, the Proximity-1 link layer operates on a single aggregated isochronous traffic stream, while the appropriate prioritizations and differentiations for individual application traffic streams are performed by the transport and network layer protocols as reflected by the ordering of these data stream *within* the aggregated isochronous at the network-link interface. It is the jitter performance over this aggregated isochronous traffic that is the focus of our analysis.

III. Prioritization Mechanism in Proximity-1

The Proximity-1 protocol is a point-to-point link layer protocol supporting expedited and sequence-controlled modes. The sequence-controlled mode has stronger reliability using a Command Operations Procedure – Proximity, (COP-P) mechanism, which essentially implements a go-back-n ARQ. The expedited mode is a best effort service in terms reliability but with higher priority to access the channel, therefore lower delay. The highest priority however is actually given to internal protocol controls such as acknowledgements or directives. The three types of frames, directive/acknowledgement, expedited, and sequence-controlled obtain access to the physical channel via a strict priority scheduler, as depicted in figure 5. The directive/acknowledgements have the highest priority, expedited has second priority, the sequence-controlled frames has the lowest priority to the channel. One does not queue any ACK frames, instead, at most a single ACK frame is maintain with the latest information on the current state of the ARQ process.


COP-P State: BOOL Need_Ack
F
T
ACK
Expedited Frame
Sequence-Controlled Frame
Strict Priority Scheduler
Figure 5: Strict Priority Scheduling in Proximity-1

Nominally, isochronous data is carried by the expedited frames for lower latency. When channel is lightly loaded, the expedited frames essentially see an open channel with minimal delay introduced by waiting for an on-going transmission, either data frame or ACK/directives to finish. The maximum delay is the transmission time (T_{tx}) of a single frame before re-gain access to the channel. Let us for a moment assume that lunar exploration will require link capacity on the order of 6Mbps, the average requirement for supporting standard definition motion imagery, then the blocking time of the channel is at most 2.7 msec.[†] which should be adequate for any real-time operation where the time constant of the control loop may range from 125msec (Low Earth Orbit) to 3+ seconds, the round-trip delay between Earth and moon. If sequence-controlled frames are not present on the forward link, the jitter is essentially the duration of a single ACK frame, which is 18.6μsec.


COP-P State: BOOL Need_Ack
In-coming sequence-controlled frames
1
2
3
4
F
T
Isochronous Traffic (expedited frames)
ack2
ack1
Strict Priority Scheduler
Out-going ACK and expedited frames
Figure 6: Isochronous traffic sharing the proximity-1 channel with acknowledgement frames

The presence of sequence-controlled frame increases the maximum jitter seen by the expedited frame by a constant factor since it can only block a freshly arrived expedited frame by at most the transmission time of one data frame. Therefore, the more dynamic aspect of the prioritization process only occurs between the expedited traffic and the directive/ACK traffic. Figure 6 shows this scenario which will be the focus of our analysis.

[†] The maximum data frame size, including synchronization marker and FEC parity, is 16440 bits.

Intuitively, one expects the system to behave in the following manner. If the rate of the in-coming sequence-controlled frames on the opposite direction is low, then the isochronous traffic will experience some but minimal channel blocking, especially when the ACK frames are much smaller than regular data frames. As the in-coming sequence control frame rate goes up, the ACK rate will increase to the point where the isochronous traffic will be blocked by a burst of ACK frames, transmitted in succession due to the time delay created by the contention. In the coming section, we will analyze the Proximity-1 protocol jitter performance under such scenario where there is a mixture of expedited traffic in one direction and acknowledgement traffic and identify circumstances under which mitigation strategies or even modifications to the specification is required.

IV. Performance of Isochronous Service using Current Proximity-1 Standard

A. Contention Process between the Data and ACK frames

We begin by examining in more details the contention process between expedited data and the acknowledgements. For easy of explanation, we refer to the direction of the isochronous traffic and acknowledgement frames as the *forward* link, and in-coming sequence-controlled frames as being carried on the *return* link. Two types of delays are distinguished: blocking and queuing. *Blocking* delay is the time duration measured from the moment a data frame reached the head of the queue, to the moment the frame actually gains access to the channel, i.e., the time instance when transmission begins. In this study, we will mostly focus our analysis on the blocking delay. Queuing delay is not discussed here since for real-time data buffering is purposely limited to reduce latency.

Keeping the strict prioritization scheme in the current Proximity-1 standard, the loading of the ACK traffic on the forward link is determined by the transition rate of the "Need_ACK" Boolean variable. The Need_ACK state is set to *True* whenever a change occurs in the state of the ARQ algorithm, and it is reset to *False* whenever a PLCW, the frame that carries the ARQ acknowledgement, is sent over the channel. So the Need_ACK variable, in a nutshell, keeps track of whether the latest ARQ state has been conveyed to the other side. Nominally, the rate of ARQ state change is driven by the arrival rate of new sequence-controlled frames on the return link (each new data frame, if received correctly in sequence, will advance the ARQ window by 1.) If the forward link is dedicated to the transmission ACK frames (as shown in Figure 7) and has sufficient capacity, then the departure rate of the ACK frames will equal to the arrival rate of data frames.

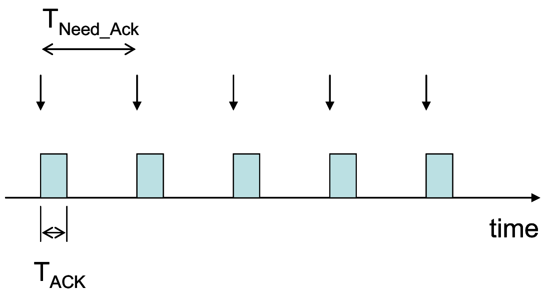


Figure 7: ACK Frame Transmissions without Contention

Let T_{Need_Ack} be the interval between successive transitions of the Need_ACK variable into the *True* state and let T_{ACK} be the transmission time of the ACK frame.

$$T_{Need\_Ack} = \frac{N_{pltu\_ret}}{R_{return}} \quad , \quad T_{Ack} = \frac{N_{plcw}}{R_{forward}} \tag{1}$$

N_{PLTU_ret}, N_{PLCW}, R_{return}, and $R_{forward}$ are the size of the data frame of the return link, size of the ACK frame, the data rate of the return link and the data rate of the forward link, respectively. The normalized loading, or duty cycle, of the ACK traffic is:

$$\rho_{ack} = \frac{T_{Ack}}{T_{Need\_ACK}} = \frac{N_{plcw}}{N_{pltu\_ret}} \cdot \frac{R_{return}}{R_{forward}} \tag{2}$$

Whenever an isochronous frame arrives to the head of the queue for transmission, it may encounter blocking when there is an on-going transmission. Clearly the larger the data frame size or forward link data rate, the lower the duty cycle, or ρ_{ack}, will be and the channel is more available to carry data traffic in the forward direction.

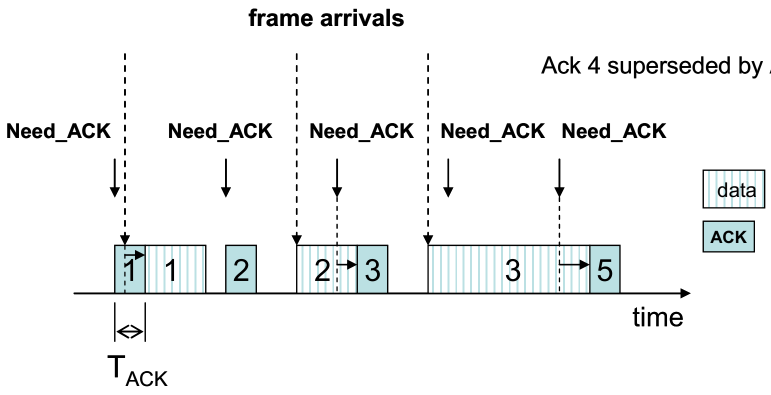


Figure 8: Contention situations between ACK and data frame

Figure 8 illustrates various contention situations between the ACK frames and the data frames. The frame #1 arrived to the head of the queue while an ACK frame is being transmitted, so it is blocked until the ACK frame clears the channel, i.e., finished transmissionl. The frame #2 arrives to an available channel, but its transmission time extends beyond the time instant when the Need_ACK #3 was triggered, so the ACK frame #3 is blocked until the data frame clears the channel. The data frame #3 arrives before Need_ACK #4 is triggered but its transmission time extends beyond Need_ACK #5. Because the Proximity-1 specification requires that all acknowledgements should sent out only the most up-to-date information regarding the ARQ process, ACK #4 is replaced by ACK#5 will be send after the data frame clears the channel. The performance of the data frame transmissions, therefore, depend on which type of contention situation is the most dominant, as a function of the loading on the forward link. Let ρ_{data} denote the offered load on the forward link isochronous traffic.

$$\rho_{data} = T_{data\_frame} \cdot \lambda_{data\_frame} = \frac{N_{pltu\_for}}{R_{forward}} \cdot \lambda_{data\_frame} \tag{3}$$

T_{data_frame} and λ_{data_frame} are the transmission time of the data frame and the offered rate of the data frame on the forward link. Note that $1/T_{data\_frame} < \lambda_{data\_frame}$. The stability of the forward link requires that

$$\rho_{data} < 1 - \rho_{ACK} = 1 - \frac{N_{plcw}}{N_{pltu\_ret}} \frac{R_{return}}{R_{forward}} \tag{4}$$

The ratio N_{plcw}/N_{pltu_ret} is the ratio of the ACK frame size to the return link data frame size. If the maximum payload size is used, it is approximately 0.0068. The typical return link to forward link data rate ratio, for the Mars Exploration Rover and Odyssey orbiter, is about 128kbps to 8kbps, a factor of 16. This means that the normalized loading of the forward link data frame should stay below 89% of the forward link bandwidth for MER-Odyssey. To keep delay and jitter under control, one should provision some margin in the design to reduce the contention.

B. Performance under light loading conditions

We begin by considering what will happen under light loading conditions, i.e., $\rho_{data} << 1 - \rho_{ack}$. Specifically, we define light loading to mean that the inter-arrival times between consecutive data frames on the forward link are large such that the latency experienced by each data frame is independent. In other words, the contention process for any data frame does not create residual effects on the next data frame. Under such condition, we only need to examine the impact of one data frame on the system to understand the average behavior over time.

1. Jitter

In the worst case scenario, a data frame arrives to the head of the queue just as an ACK frame begins transmission. At the end of the ACK frame transmission, the channel will free up because $\rho_{ack} < 1$. So the data frame can re-gain access. Therefore, the maximum blocking delay jitter under light loading condition is just T_{ACK}, which is the maximum latency a data frame will experience. Even though the data frame arrival rate is low, the size of the data frame may have a significant impact on the ARQ process for the return link. Here we have to consider how contention may delay and even create gaps in the ACK frame stream.

2. Impact on ARQ: Short Data Frame on Forward Link

If the transmission time $T_{data\_frame} \leq T_{Need\_Ack}$, any data frame transmission cannot overlap two consecutive resets of the Need_ACK state. Therefore, no ACK frames will be superseded by later ACKs due to extended blocking, as shown in earlier examples, by a data frame. One expects that the throughput of the ARQ process on the return link data frame will not be impacted significantly. The effect of the acknowledgement jitter on the ARQ process can be neutralized by simply increasing the window size by one, therefore accommodating the maximum jitter without triggering unnecessary retransmissions.

3. Impact on ARQ: Long Data Frame on Forward Link

However, if the data frame on the forward link is large such that $T_{data\_frame} > n * T_{Need\_Ack}$ where n is positive integer, then each data frame *could* potentially block n + 1 ACKs, and causing n ACKs to be skipped. Figure 8 illustrate one such case. In other words, the ACK for data frame #m on the return link will, after some delay, be followed by the ACK for data frame #(m+n-1) on the return link. The ARQ algorithm will continue to operate correctly under such situation because the protocol recognizes cumulative acknowledgement, i.e., by acknowledging frame #m, all frame prior to frame #m are implicitly acknowledged. However, there will be some loss of efficiency if the protocol is not configured to take this into account and execute re-transmissions that are not necessary.

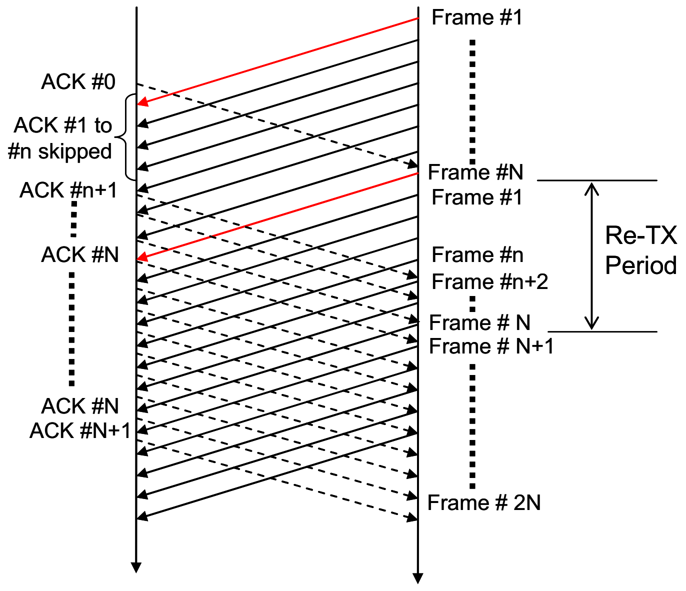


Figure 9: ARQ with ACK gap n < N

Under nominal conditions, the window size of the ARQ protocol is set to a specific value N to ensure that before frame #N completes transmission the acknowledgement of frame #1 should be received. N is typically selected, depending on the round-trip latency, such that the RF channel is kept at its maximum utilization with new frames being transmitted continuously without 'dead air.' However, N is usually selected under the assumption there is a continuous, gap-free reception of acknowledgement frame – a reasonable assumption when forward link is reliable and dedicated to carrying ACK frames only. If there is contention between the ACK and data frame, such typical selection of N may cause pre-mature re-transmission.

We define a re-tx period as illustrated in Figure 9 in which we assumed that $n \leq N-1$. The period begins when at the end of sending frame #N, the sender did not received the expected acknowledgement for frame #1, so it goes into a “progressive retransmission process” by sending frame #1 again assuming that frame #1 to #N were lost. After missing n ACKs, ACK for frame #n+1 is finally received by the sender, at which time the sender has just finished re-transmitting frame #n and will jump from sending frame #n to #n+2, having received the ACK for frame #n+1. The cycle completes when the sender starts the transmission of frame #N+1. All together N-1 frames were re-transmitted unnecessarily, therefore we have $T_{re\_tx\_period} = N_{pltu}/R_{return}$ (N-1).

One can further extend the analysis to situations where n > N-1. In that case (see Figure 10) when the ACKs finally resumed, it will acknowledge frame #N because that is the highest ever observed by the receiver, and the sender side will immediately jump right up to transmit frame # N+1, for a total of n duplicate transmissions.

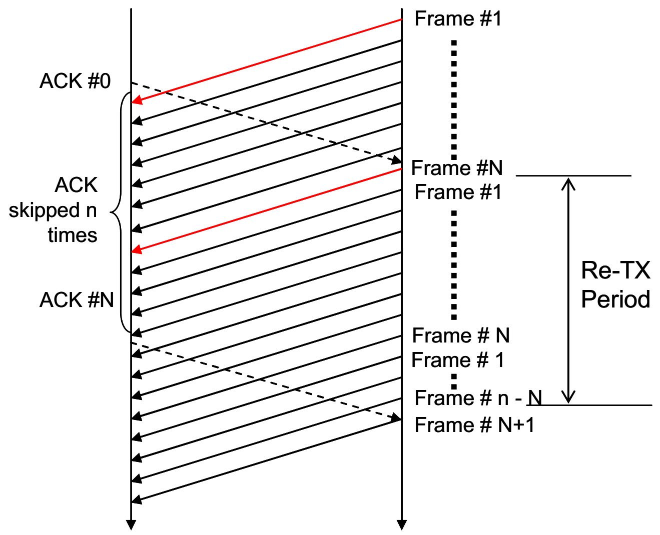


Figure 10: ARQ with ACK gap cycle (n ≥ N)

We have examined the impact of one ACK gap caused by a single data frame on the forward link because under light loading conditions where the inter-arrival time of the data frames are sufficiently large; in other words, $1/\lambda_{data\_frame} > \max(n,N-1) * T_{Need\_ACK} = \max(n,N-1) * N_{pltu\_ret}/R_{return}$. Since we know that n, the gap size in the ACK frames, is the greatest integer less than the ratio of the transmission time of the forward link data frame and the Need_ACK period, which is given by $\lfloor T_{data\_frame}/T_{Need\_ACK} \rfloor$. Then the efficiency of the ARQ process, or η_{ARQ}, when $T_{data\_frame} > T_{Need\_ACK}$, is given by:

$$\begin{aligned}\eta_{ARQ} &= 1-\lambda_{data\_frame}\cdot\max\left(n, N-1\right)\cdot\frac{N_{pltu\_ret}}{R_{return}} \\ &= 1-\lambda_{data\_frame}\cdot\max\left(\left\lfloor\frac{N_{pltu\_for}\cdot R_{return}}{N_{pltu\_ret}\cdot R_{forward}}\right\rfloor, N-1\right)\cdot\frac{N_{pltu\_ret}}{R_{return}}\end{aligned} \tag{5}$$

Typically N is chosen to be the least number of return frames that will fill the round-trip time, which is $\lceil T_{round\text{-}trip}/(N_{pltu\_ret}/R_{return})\rceil$, The round trip time $T_{round\text{-}trip}$ includes the two-way propagation time, transmission, coding, and processing delays of the data frame on the return link and ACK frame on the forward link. Substituting this into equation (5), we can compute the expected ARQ efficiency under light loading condition.

$$\eta_{ARQ} = 1-\lambda_{data\_frame}\cdot\max\left(\left\lfloor\frac{N_{pltu\_for}\cdot R_{return}}{N_{pltu\_ret}\cdot R_{forward}}\right\rfloor, \left\lceil\frac{T_{round-trip}\cdot R_{return}}{N_{pltu\_ret}}\right\rceil - 1\right)\cdot\frac{N_{pltu\_ret}}{R_{return}} \tag{6}$$

4. *General Recommendations under light loading conditions*

The jitter experienced by forward link frame is T_{ack} under light loading conditions, which is a function of the forward link data rate. Because contention is actually low, the delay jitter will not be improved by giving isochronous data frame higher access priority than the ACK frames since the worst case blocking occurs when the ACK already begins transmission, at which point the channel is already committed.

As equation (5) indicates, the ARQ efficiency is driven by the round-trip time as well as the size transmission time of the forward link data frames. To minimize loss in ARQ efficiency, one can either: (a) increase the window size to N + n, where n is the maximum number of ACKs that could be skipped due to blocking by a forward link frame, to prevent pre-mature re-transmissions, or (b) reduce the forward link frame size so that $T_{data\_frame} < T_{Need\_ACK}$; this will eliminate the creation of ACK gaps.

In general, under light loading conditions there is no need to modify the current prioritization scheme in the Proximity-1 specification. However, if the required jitter cannot be met by increasing forward link bandwidth, one may modify the current standard to include a weighted fair scheduler so that data frames will receive guaranteed bandwidth at the expense of creating ACK gaps. One can then mitigate the impact of the ACK gap by increasing the ARQ window size appropriately.

C. Performance under high loading conditions

High loading condition means that the offered load of the forward link data, ρ_{data} is significant compare to 1 - ρ_{ack}, which means that the contention between data and ACK frames become very intense and tend to correlated from one data frame to the next. The effect of this contention will create more channel blocking and higher jitter, which is detrimental to the isochronous traffic. To derive the upper bound on delay jitter, the worst case blocking by the ACK frames will be analyzed. The current Proximity-1 specification requires that only latest information should be sent in ARQ acknowledgement message. Therefore, when the ARQ state is updated, any buffered ACK frame *waiting* for transmission will be replaced by a new ACK frame. Because only the latest ACK frame is queued for transmission, long blockage due to ACK frame transmission normally does not occur. The only scenario in which a data frame could be blocked by multiple, consecutive ACK frames is when the ARQ state is updated *while* an ACK frame is being transmitted. Then at the end of the ACK frame transmission, the channel will not be released to the data frame but immediately used to send another updated ACK. This is the focus of our analysis here.

Assuming that $\rho_{ack} < 1$, the ARQ state will not change on top of an on-going ACK frame transmission unless the ACK frame being transmitted were *delayed* by a data frame. Two cases can be further distinguished and considered: (1) $\rho_{ack} < 1/2$ and (2) $1 > \rho_{ack} \geq 1/2$.

1. Case 1: $\rho_{ack} < 1/2$

Figure 11 illustrates one such scenario. If an ACK frame is delayed sufficiently such that during its transmission time, the ARQ state changed again, then at the end of the ACK frame transmission, another ACK frame will follow immediately. Under high loading condition, we assume that the 2nd data frame could potentially arrive and move immediately to the head of the queue within the window of 2 * T_{ack}. Then the 2nd data frame will not gain channel access until the end of the 2nd ACK frame transmission. Because $\rho_{ack} < 1/2$, (or equivalently, $T_{Need\_Ack} > 2 * T_{Ack}$) there is no danger of having the second ACK frame blocking the third ACK frame. This means that number of consecutive ACK frames transmissions is upper-limited to two, thus the maximum delay jitter due to channel contention is 2 * T_{ack}, doubling the bound under light loading condition.

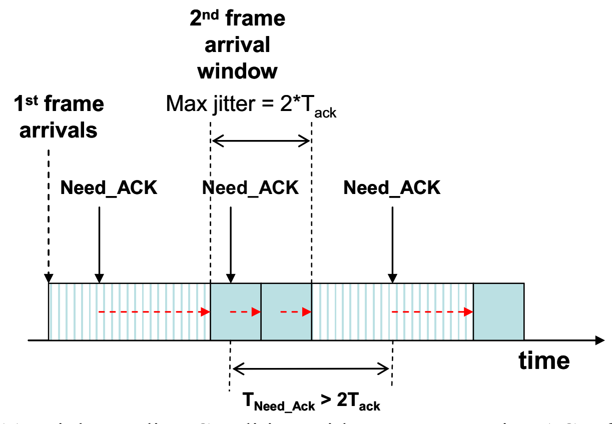


Figure 11: High Loading Condition with two consecutive ACK frames

When the loading of the data frame approach 1-ρ_{ack}, significant queuing delay, as well as contention (blocking delay) will drive the latency and jitter performance and degrade the quality of service. However, in reality, when such situation occurs, most system will purposely allow data overflow to keep make sure that once the congestion is over, the newly arrived data frames are not queued behind outdated frames.

2. *Case 2: $1 > \rho_{ack} \geq 1/2$*

It is also possible that a stream of k (>2) ACK frames may occur, when ρ_{ack} is sufficiently high as shown in Figure 12.

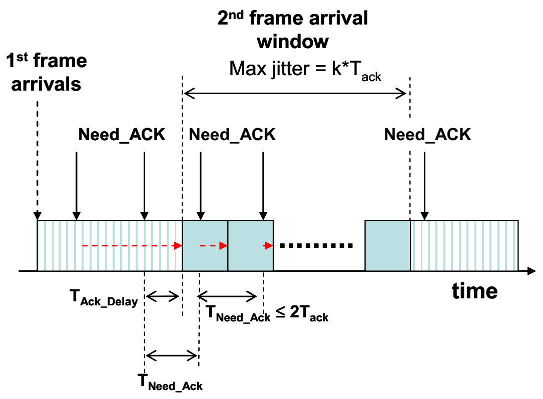


Figure 12: Channel Blocked by K Consecutive ACK Frames

To compute the number of consecutive ACK frames that may occur, we define $T_{def}(i+1)$ as the delay experienced by the (i+1)th ACK frame, due to blocking by the i-th ACK frame. Note that by definition T_{def} is always less than T_{Need_Ack} because only the latest ACK frame is buffered for transmission. Figure 13 shows the timing relationship.

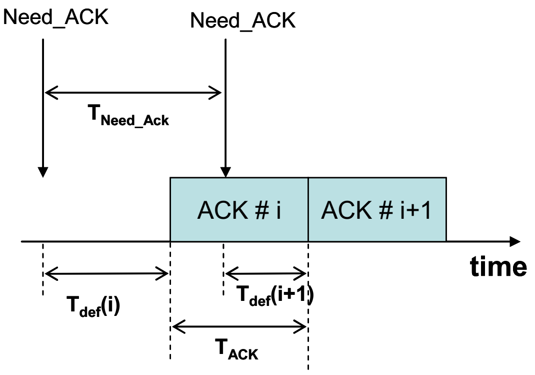


Figure 13: $T_{def}(i)$ Timing Diagram

The iterative relationship between $T_{def}(i)$ is given by:

$$T_{def}(i+1) = T_{ACK} - T_{Need\_Ack} + T_{def}(i) \tag{6}$$

where $T_{def}(i) > 0$. Note that since $T_{Ack} < T_{Need\_Ack}$, $T_{def}(i)$ is always decreasing, and the burst length of the ACK frames is the largest index *i* such that $T_{def}(i+1) > 0$, i.e., the number of consecutive frames, K, is given by $\lceil T_{def}(1)/(T_{Need\_Ack} - T_{ACK}) \rceil$. The maximum delay jitter occurs when $T_{def}(1) = T_{Need\_ACK}$:

$$D_{\max\_jitter} = T_{ACK} \max_{i>0} \left\{ T_{def}(i+1) > 0, T_{def}(1) = T_{Need\_ACK} \right\} = T_{ACK} \left\lceil \frac{T_{Need\_ACK}}{T_{Need\_ACK} - T_{ACK}} \right\rceil = T_{ACK} \left\lceil \frac{1}{1-\rho_{ACK}} \right\rceil \tag{7}$$

This result also generalizes to the $\rho_{Ack} < 1/2$ case. As $\rho_{Ack} \rightarrow 1$, the maximum jitter becomes *unbounded*, making the forward link unsuitable for isochronous traffic. In reality what may occur in this case is that the system will simply dropped buffered isochronous packets, thus limiting the latency for later packets and frames handled by

Proximity-1. This is analogous to suffering a temporary outage while conversing on the phone. When the outage ends, instead of hearing all the “missed” words/sentences again, the conversation simply resumes with a open channel, nothing is buffered, stored, or played back; all voice data during the outage were simply dropped. In our case, if a very long burst occurs, the system should similarly drop most of the data so that when the burst of ACK ends, what flows across the link are fresh isochronous traffic.

3. *Impact on ARQ Efficiency*

The impact on ARQ efficiency on the reverse direction is similar to the light loading case. The driving factor is the transmission time of the data frame, if the data frame is sufficient large and will create ACK gaps, the window size of the ARQ algorithm should be adjusted accordingly. When the data frame is small, the jitter on ACK frame is bounded by T_{Need_Ack}, which can be easily compensated by increasing the window size by one.

D. Contention with Sequence-controlled Frames

Although not the focus of our analysis, it is possible that there are expedited frames present on the forward link. The worse case latency experienced by an expedited frame occurs when an expedited frame arrives just after a sequence-controlled frame gains access to the channel, which is then followed by one or a burst of ACK frames. In general, this scenario increases the maximum jitter derived in our previous analysis by the transmission time of one data frame. A data frame could be larger than the ACK frame by two orders of magnitude, so the increase in jitter is potentially large. However, given the strict-priority assigned to the expedited frames, this additional latency stays constant regardless of the loading of the sequence-controlled traffic. The user can limit the jitter caused by sequence-controlled frame by keeping the data frame size on the forward link small when configuring the link.

V. Additional Considerations

So far we have analyzed the maximum delay jitter for the aggregated isochronous traffic, as supported by the expedited Proximity-1 service. It is assumed that fine-grain prioritizations among multiple isochronous data streams generated by the various real-time applications were handled by the higher layers of the protocol stack. For example, the network and transport layers can control the order in which real-time packets from various applications were multiplexed serially into the Proximity-1 link layer interface. In this manner, the transport, network and link layers all have impact on the end-to-end latency and jitter. A modification to the current specifications is required to prioritize transmissions on a per application basis.

As described in earlier section, the specification requires that whenever the channel is available for transmitting an acknowledgement frame, the latest ARQ state information should be reflected. In other words, the protocol should at any time instance buffer at most one ACK frame whose information content reflects the latest ARQ state. If while the ACK frame is waiting for transmission the ARQ state changed, then either the ACK frame buffered should be replaced with a new one. One can think of the ACK frame buffering mechanism as a last-in first-out queue that holds only one frame. However, given that most of the time the ACK frame loading is low, the need to replace an already buffered ACK frame is fairly small, this may result in implementations that actually maintains a small queue of ACK frames and increased jitter under the high loading scenarios, which is not analyzed in this study.

Queuing delay that occurs as a result of temporarily traffic fluctuation has not been considered here although it is clear that some buffering is needed at the network-link layer interface. Queuing delay is a strong function of the inter-arrival time distribution of the traffic, the capacity of the network, and variance of the service time. It is usually somewhat mitigated by the proper provisioning of system resources and implementation of flow/congestion control on the application and transport layers. For real-time traffic streams, it is preferred that isochronous packets/frames be dropped when there is congestion, outage, or long blocking, instead of buffering them for delayed transmission; therefore we do not address it in this analysis.

Finally, it should be noted that this analysis did not assume any particular coding or physical layer implementation and therefore would apply if the Proximity-1 data link layer is applied on top of alternative coding or physical layer standards.

VI. Recommendations and Conclusions

In this study, we analyzed the jitter performance of the CCSDS Proximity-1 Space Link protocol in the context of providing isochronous quality of service. We presented analysis of the maximum jitter of the protocol based on the current specification and showed that it is an inverse function of the residual bandwidth on the forward link, i.e., $1-\rho_{Ack}$. The presence of sequence-controlled frames on the forward link will further increase the maximum jitter by the transmission time of a sequence-controlled data frame, which can be limited by user configuration of the link. The impact of providing isochronous service on ARQ traffic on the return link can be minimized when the ARQ window size is properly configured. In this study we have assumed that the link layer operates over an aggregated stream of isochronous traffic and does not prioritize transmissions for individual isochronous applications flows, that functionality is assumed to be performed by higher layers of the protocol stack which has visibility to metadata associated with each isochronous data unit.

In general, it is not necessary to modify the current specification for isochronous services under light loading conditions. Voice-Over-IP (VoIP) streams require from 8 up to 64kbps throughput (depending on the codec selected), which should be easily supported by future radios with high forward link capacity. Current specification of the Proximity-1 protocol provide an Optional Control Field (OCF) in the frame structure which can piggy-back acknowledgement information in the data stream to reduce the loading of the acknowledge traffic as well. For scenarios where (1) the acknowledgement traffic is dominant on the forward link, (2) jitter requirement is tighter than the transmission time of an acknowledgement/data frame, or (3) prioritization of individual isochronous application flows is desired at the link layer, then alternative mechanisms that dynamically adjust the priority of each frame should be considered.

Acknowledgments

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

References

[1] "CCSDS Proximity-1 Space Link Protocol — Data Link Layer," Blue Book, Issue 3, May 2004, URL: http://public.ccsds.org/publications/archive/211x0b3.pdf [cited March 23, 2006]

[2] ITU-T Recommendation G.114, "One-way Transmission Time"